**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

Санкт-Петербургский национальный исследовательский университет

информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Проектирование баз данных**

**Лабораторная работа №2.**

Создание таблиц в PostgresSQL

Вариант №6 (Библиотека)

Выполнил студент группы №М34061

**Величко Максим**

Преподаватель:

**Боркунов Владимир Юрьевич**

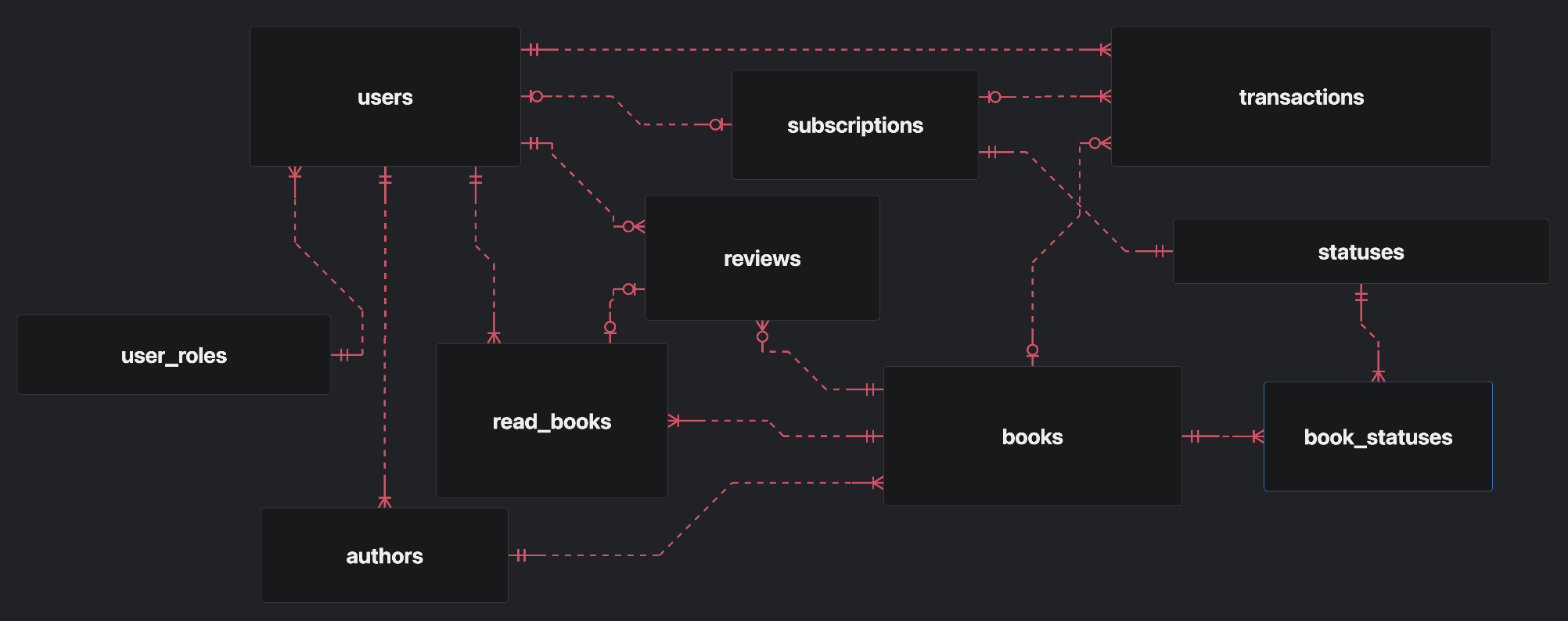
САНКТ-ПЕТЕРБУРГ

2024

**Краткое описание предметной области**

Сайт "ЛитРес" — это онлайн-библиотека, предлагающая пользователям доступ к электронным книгам, аудиокнигам и другим публикациям. Пользователи могут просматривать, читать и оставлять отзывы о книгах, а также управлять своими подписками и любимыми книгами. Модель данных включает в себя все ключевые аспекты, необходимые для работы сайта: информация о книгах, авторах, пользователях, транзакциях и статусах книг.

**Изображение выглядит как снимок экрана, диаграмма, текст

Автоматически созданное описаниеСтруктура базы данных**

**Изображение выглядит как снимок экрана, текст, диаграмма, дизайн

Автоматически созданное описание**

**Скрипт для формирования таблиц:**

DROP TABLE IF EXISTS read\_books CASCADE;

DROP TABLE IF EXISTS transactions CASCADE;

DROP TABLE IF EXISTS subscriptions CASCADE;

DROP TABLE IF EXISTS reviews CASCADE;

DROP TABLE IF EXISTS book\_statuses CASCADE;

DROP TABLE IF EXISTS books CASCADE;

DROP TABLE IF EXISTS authors CASCADE;

DROP TABLE IF EXISTS user\_roles CASCADE;

DROP TABLE IF EXISTS statuses CASCADE;

DROP TABLE IF EXISTS users CASCADE;

CREATE TABLE statuses

(

id uuid NOT NULL PRIMARY KEY,

name VARCHAR(255) NOT NULL

);

CREATE TABLE user\_roles

(

id uuid NOT NULL PRIMARY KEY,

role INT NOT NULL DEFAULT 1 CHECK (role IN (1, 2, 3)) -- 1 - reader, 2 - author, 3 - admin

);

CREATE TABLE subscriptions

(

id uuid NOT NULL PRIMARY KEY,

type VARCHAR(255) NOT NULL DEFAULT 'standard',

start\_date TIMESTAMP NOT NULL,

end\_date TIMESTAMP NOT NULL,

status\_id uuid NOT NULL,

CONSTRAINT FK\_statuses\_TO\_subscriptions FOREIGN KEY (status\_id) REFERENCES statuses (id) ON DELETE CASCADE

);

CREATE TABLE users

(

id uuid NOT NULL PRIMARY KEY,

name VARCHAR(255) NOT NULL UNIQUE,

email VARCHAR(255) NOT NULL UNIQUE CHECK (email LIKE '%\_@\_\_%.\_\_%'),

password VARCHAR(255) NOT NULL,

registration\_date TIMESTAMP NOT NULL,

subscription\_id uuid,

user\_role\_id uuid NOT NULL,

CONSTRAINT FK\_subscriptions\_TO\_users FOREIGN KEY (subscription\_id) REFERENCES subscriptions (id) ON DELETE SET NULL,

CONSTRAINT FK\_user\_roles\_TO\_users FOREIGN KEY (user\_role\_id) REFERENCES user\_roles (id) ON DELETE CASCADE

);

CREATE TABLE authors

(

id uuid NOT NULL PRIMARY KEY,

name VARCHAR(255) NOT NULL UNIQUE,

bio VARCHAR(511),

user\_id uuid NOT NULL,

CONSTRAINT FK\_users\_TO\_authors FOREIGN KEY (user\_id) REFERENCES users (id) ON DELETE CASCADE

);

CREATE TABLE books

(

id uuid NOT NULL PRIMARY KEY,

title VARCHAR(255) NOT NULL UNIQUE,

year INTEGER, -- Changed YEAR to INTEGER

price DECIMAL(10, 2) DEFAULT 0 CHECK (price >= 0),

description TEXT,

genre INT NOT NULL CHECK (genre IN (1, 2, 3, 4, 5)),

author\_id uuid NOT NULL,

CONSTRAINT FK\_authors\_TO\_books FOREIGN KEY (author\_id) REFERENCES authors (id) ON DELETE CASCADE

);

CREATE TABLE book\_statuses

(

id uuid NOT NULL PRIMARY KEY,

start\_date TIMESTAMP NOT NULL,

end\_date TIMESTAMP NOT NULL,

book\_id uuid NOT NULL,

status\_id uuid NOT NULL,

CONSTRAINT FK\_books\_TO\_book\_statuses FOREIGN KEY (book\_id) REFERENCES books (id) ON DELETE CASCADE,

CONSTRAINT FK\_statuses\_TO\_book\_statuses FOREIGN KEY (status\_id) REFERENCES statuses (id) ON DELETE CASCADE

);

CREATE TABLE reviews

(

id uuid NOT NULL PRIMARY KEY,

rating INT NOT NULL CHECK (rating >= 1 AND rating <= 5),

text TEXT,

date TIMESTAMP NOT NULL,

user\_id uuid NOT NULL,

book\_id uuid NOT NULL,

CONSTRAINT FK\_users\_TO\_reviews FOREIGN KEY (user\_id) REFERENCES users (id) ON DELETE CASCADE,

CONSTRAINT FK\_books\_TO\_reviews FOREIGN KEY (book\_id) REFERENCES books (id) ON DELETE CASCADE

);

CREATE TABLE read\_books

(

id uuid NOT NULL PRIMARY KEY,

start\_date TIMESTAMP,

end\_date TIMESTAMP,

pagesRead INT DEFAULT 0 CHECK (pagesRead >= 0),

totalPages INT NOT NULL CHECK (totalPages > 0),

user\_id uuid NOT NULL,

book\_id uuid NOT NULL,

review\_id uuid,

CONSTRAINT FK\_users\_TO\_read\_books FOREIGN KEY (user\_id) REFERENCES users (id) ON DELETE CASCADE,

CONSTRAINT FK\_books\_TO\_read\_books FOREIGN KEY (book\_id) REFERENCES books (id) ON DELETE CASCADE,

CONSTRAINT FK\_reviews\_TO\_read\_books FOREIGN KEY (review\_id) REFERENCES reviews (id) ON DELETE CASCADE

);

CREATE TABLE transactions

(

id uuid NOT NULL PRIMARY KEY,

transaction\_kind INT NOT NULL CHECK (transaction\_kind IN (1, 2, 3)),

date TIMESTAMP NOT NULL,

cost DECIMAL(10, 2) DEFAULT 0 CHECK (cost >= 0),

user\_id uuid NOT NULL,

book\_id uuid,

subscription\_id uuid,

CONSTRAINT FK\_users\_TO\_transactions FOREIGN KEY (user\_id) REFERENCES users (id) ON DELETE CASCADE,

CONSTRAINT FK\_books\_TO\_transactions FOREIGN KEY (book\_id) REFERENCES books (id) ON DELETE CASCADE,

CONSTRAINT FK\_subscriptions\_TO\_transactions FOREIGN KEY (subscription\_id) REFERENCES subscriptions (id) ON DELETE CASCADE

);

CREATE UNIQUE INDEX IF NOT EXISTS authors\_name ON authors(name);

CREATE UNIQUE INDEX IF NOT EXISTS title ON books(title);

CREATE UNIQUE INDEX IF NOT EXISTS email ON users(email);

**Скрипт для заполнения данных:**

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

-- 1. Заполнение таблицы статусов

INSERT INTO statuses (id, name) VALUES

(uuid\_generate\_v4(), 'Active'),

(uuid\_generate\_v4(), 'Inactive'),

(uuid\_generate\_v4(), 'Pending'),

(uuid\_generate\_v4(), 'Cancelled'),

(uuid\_generate\_v4(), 'Expired');

-- 2. Заполнение таблицы ролей пользователей

INSERT INTO user\_roles (id, role) VALUES

(uuid\_generate\_v4(), 1), -- Reader

(uuid\_generate\_v4(), 2), -- Author

(uuid\_generate\_v4(), 3); -- Admin

-- 3. Заполнение таблицы подписок

INSERT INTO subscriptions (id, type, start\_date, end\_date, status\_id) VALUES

(uuid\_generate\_v4(), 'Standard', NOW(), NOW() + INTERVAL '1 year', (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), 'Premium', NOW(), NOW() + INTERVAL '1 year', (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), 'Basic', NOW(), NOW() + INTERVAL '1 year', (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), 'Family', NOW(), NOW() + INTERVAL '1 year', (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), 'Student', NOW(), NOW() + INTERVAL '1 year', (SELECT id FROM statuses WHERE name='Pending'));

-- 4. Заполнение таблицы пользователей

INSERT INTO users (id, name, email, password, registration\_date, subscription\_id, user\_role\_id) VALUES

(uuid\_generate\_v4(), 'Alice', 'alice@example.com', 'password1', NOW(), (SELECT id FROM subscriptions WHERE type = 'Standard'), (SELECT id FROM user\_roles WHERE role = 1)),

(uuid\_generate\_v4(), 'Bob', 'bob@example.com', 'password2', NOW(), (SELECT id FROM subscriptions WHERE type = 'Premium'), (SELECT id FROM user\_roles WHERE role = 2)),

(uuid\_generate\_v4(), 'Charlie', 'charlie@example.com', 'password3', NOW(), (SELECT id FROM subscriptions WHERE type = 'Basic'), (SELECT id FROM user\_roles WHERE role = 3)),

(uuid\_generate\_v4(), 'David', 'david@example.com', 'password4', NOW(), (SELECT id FROM subscriptions WHERE type = 'Family'), (SELECT id FROM user\_roles WHERE role = 1)),

(uuid\_generate\_v4(), 'Eva', 'eva@example.com', 'password5', NOW(), (SELECT id FROM subscriptions WHERE type = 'Student'), (SELECT id FROM user\_roles WHERE role = 2)),

(uuid\_generate\_v4(), 'Frank', 'frank@example.com', 'password6', NOW(), (SELECT id FROM subscriptions WHERE type = 'Standard'), (SELECT id FROM user\_roles WHERE role = 1)),

(uuid\_generate\_v4(), 'Grace', 'grace@example.com', 'password7', NOW(), (SELECT id FROM subscriptions WHERE type = 'Premium'), (SELECT id FROM user\_roles WHERE role = 2)),

(uuid\_generate\_v4(), 'Hannah', 'hannah@example.com', 'password8', NOW(), (SELECT id FROM subscriptions WHERE type = 'Basic'), (SELECT id FROM user\_roles WHERE role = 1)),

(uuid\_generate\_v4(), 'Ian', 'ian@example.com', 'password9', NOW(), (SELECT id FROM subscriptions WHERE type = 'Family'), (SELECT id FROM user\_roles WHERE role = 3)),

(uuid\_generate\_v4(), 'Jack', 'jack@example.com', 'password10', NOW(), (SELECT id FROM subscriptions WHERE type = 'Student'), (SELECT id FROM user\_roles WHERE role = 1));

-- 5. Заполнение таблицы авторов

INSERT INTO authors (id, name, bio, user\_id) VALUES

(uuid\_generate\_v4(), 'Author One', 'Bio of Author One', (SELECT id FROM users WHERE name='Bob')),

(uuid\_generate\_v4(), 'Author Two', 'Bio of Author Two', (SELECT id FROM users WHERE name='Eva')),

(uuid\_generate\_v4(), 'Author Three', 'Bio of Author Three', (SELECT id FROM users WHERE name='Charlie'));

-- 6. Заполнение таблицы книг

INSERT INTO books (id, title, year, price, description, genre, author\_id) VALUES

(uuid\_generate\_v4(), 'The Great Gatsby', 1925, 10.99, 'A classic novel.', 1, (SELECT id FROM authors WHERE name='Author One')),

(uuid\_generate\_v4(), '1984', 1949, 8.99, 'A dystopian novel.', 2, (SELECT id FROM authors WHERE name='Author Two')),

(uuid\_generate\_v4(), 'To Kill a Mockingbird', 1960, 9.99, 'A novel about racial inequality.', 3, (SELECT id FROM authors WHERE name='Author Three')),

(uuid\_generate\_v4(), 'Pride and Prejudice', 1813, 12.99, 'A romantic novel.', 4, (SELECT id FROM authors WHERE name='Author One')),

(uuid\_generate\_v4(), 'Moby-Dick', 1851, 11.99, 'An adventure novel.', 5, (SELECT id FROM authors WHERE name='Author Three'));

-- 7. Заполнение таблицы статусов книг

INSERT INTO book\_statuses (id, start\_date, end\_date, book\_id, status\_id) VALUES

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '30 days', (SELECT id FROM books WHERE title='The Great Gatsby'), (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '60 days', (SELECT id FROM books WHERE title='1984'), (SELECT id FROM statuses WHERE name='Pending')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '45 days', (SELECT id FROM books WHERE title='To Kill a Mockingbird'), (SELECT id FROM statuses WHERE name='Expired')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '90 days', (SELECT id FROM books WHERE title='Pride and Prejudice'), (SELECT id FROM statuses WHERE name='Active')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '75 days', (SELECT id FROM books WHERE title='Moby-Dick'), (SELECT id FROM statuses WHERE name='Cancelled'));

-- 8. Заполнение таблицы рецензий

INSERT INTO reviews (id, rating, text, date, user\_id, book\_id) VALUES

(uuid\_generate\_v4(), 5, 'Great book!', NOW(), (SELECT id FROM users WHERE name='Alice'), (SELECT id FROM books WHERE title='The Great Gatsby')),

(uuid\_generate\_v4(), 4, 'Very interesting.', NOW(), (SELECT id FROM users WHERE name='Bob'), (SELECT id FROM books WHERE title='1984')),

(uuid\_generate\_v4(), 3, 'It was okay.', NOW(), (SELECT id FROM users WHERE name='Charlie'), (SELECT id FROM books WHERE title='To Kill a Mockingbird')),

(uuid\_generate\_v4(), 5, 'Loved it!', NOW(), (SELECT id FROM users WHERE name='David'), (SELECT id FROM books WHERE title='Pride and Prejudice')),

(uuid\_generate\_v4(), 2, 'Not my taste.', NOW(), (SELECT id FROM users WHERE name='Eva'), (SELECT id FROM books WHERE title='Moby-Dick'));

-- 9. Заполнение таблицы прочитанных книг

INSERT INTO read\_books (id, start\_date, end\_date, pagesRead, totalPages, user\_id, book\_id) VALUES

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '15 days', 100, 200, (SELECT id FROM users WHERE name='Alice'), (SELECT id FROM books WHERE title='The Great Gatsby')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '10 days', 150, 300, (SELECT id FROM users WHERE name='Bob'), (SELECT id FROM books WHERE title='1984')),

(uuid\_generate\_v4(), NOW(), NOW() + INTERVAL '5 days', 50, 120, (SELECT id FROM users WHERE name='Charlie'), (SELECT id FROM books WHERE title='To Kill a Mockingbird'));

-- 10. Заполнение таблицы транзакций

COPY transactions (id, transaction\_kind, date, cost, user\_id, book\_id, subscription\_id)

FROM 'transactions.csv' DELIMITER ',' CSV HEADER;

**Файл transactions.csv:**

**Изображение выглядит как текст, снимок экрана, меню, число

Автоматически созданное описание**

Данный скрипт тестировался на сервисе[**Онлайн PostgresSQL**](https://sqliteonline.com/)